

# Improved Approximation Algorithms for Maximum Cut and Satisfiability Problems Using Semidefinite Programming

MICHEL X. GOEMANS

*Massachusetts Institute of Technology, Cambridge, Massachusetts*

AND

DAVID P. WILLIAMSON

*IBM T. J. Watson Research Center, Yorktown Heights, New York*

**Abstract.** We present randomized approximation algorithms for the maximum cut (MAX CUT) and maximum 2-satisfiability (MAX 2SAT) problems that always deliver solutions of expected value at least .87856 times the optimal value. These algorithms use a simple and elegant technique that randomly rounds the solution to a nonlinear programming relaxation. This relaxation can be interpreted both as a semidefinite program and as an eigenvalue minimization problem. The best previously known approximation algorithms for these problems had performance guarantees of  $\frac{1}{2}$  for MAX CUT and  $\frac{3}{4}$  for MAX 2SAT. Slight extensions of our analysis lead to a .79607-approximation algorithm for the maximum directed cut problem (MAX DICUT) and a .758-approximation algorithm for MAX SAT, where the best previously known approximation algorithms had performance guarantees of  $\frac{1}{4}$  and  $\frac{3}{4}$ , respectively. Our algorithm gives the first substantial progress in approximating MAX CUT in nearly twenty years, and represents the first use of semidefinite programming in the design of approximation algorithms.

**Categories and Subject Descriptors:** F2.2 [Analysis of Algorithms and Problem Complexity]: Nonnumerical Algorithms and Problems—*computations on discrete structures*; G2.2 [Discrete Math-

---

A preliminary version has appeared in *Proceedings of the 26th Annual ACM Symposium on Theory of Computing* (Montreal, Que., Canada). ACM, New York, 1994, pp. 422–431.

The research of M. X. Goemans was supported in part by National Science Foundation (NSF) contract CCR 93-02476 and DARPA contract N00014-92-J-1799.

The research of D. P. Williamson was supported by an NSF Postdoctoral Fellowship. This research was conducted while the author was visiting MIT.

**Authors' addresses:** M. X. Goemans, Department of Mathematics, Room 2-382, Massachusetts Institute of Technology, Cambridge, MA 02139, e-mail: goemans@math.mit.edu; D. P. Williamson, IBM T. J. Watson Research Center, Room 33-219, P.O. Box 218, Yorktown Heights, NY 10598, e-mail: dpw@watson.ibm.com.

Permission to make digital/hard copy of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage, the copyright notice, the title of the publication, and its date appear, and notice is given that copying is by permission of ACM, Inc. To copy otherwise, to republish, to post on servers, or to redistribute to lists, requires prior specific permission and/or a fee.

© 1995 ACM 0004-5411/95/1100-1115 \$03.50

**ematics**]: Graph Theory—*graph algorithms*; G3 [Probability and Statistics]—*probabilistic algorithms (including Monte-Carlo)*; I1.2 [Algebraic manipulation]: Algorithms—*analysis of algorithms*

General Terms: Algorithms

Additional Key Words and Phrases: Approximation algorithms, convex optimization, randomized algorithms, satisfiability

## 1. Introduction

Given an undirected graph  $G = (V, E)$  and nonnegative weights  $w_{ij} = w_{ji}$  on the edges  $(i, j) \in E$ , the maximum cut problem (MAX CUT) is that of finding the set of vertices  $S$  that maximizes the weight of the edges in the *cut*  $(S, \bar{S})$ ; that is, the weight of the edges with one endpoint in  $S$  and the other in  $\bar{S}$ . For simplicity, we usually set  $w_{ij} = 0$  for  $(i, j) \notin E$  and denote the weight of a cut  $(S, \bar{S})$  by  $w(S, \bar{S}) = \sum_{i \in S, j \notin S} w_{ij}$ . The MAX CUT problem is one of the Karp's original NP-complete problems [Karp 1972] and has long been known to be NP-complete even if the problem is unweighted; that is, if  $w_{ij} = 1$  for all  $(i, j) \in E$  [Garey et al. 1976]. The MAX CUT problem is solvable in polynomial time for some special classes of graphs (e.g., if the graph is planar [Orlova and Dorfman 1972; Hadlock 1975]). Besides its theoretical importance, the MAX CUT problem has applications in circuit layout design and statistical physics (Barahona et al. [1988]). For a comprehensive survey of the MAX CUT problem, the reader is referred to Poljak and Tuza [1995].

Because it is unlikely that there exist efficient algorithms for NP-hard maximization problems, a typical approach to solving such a problem is to find a  $\rho$ -approximation algorithm; that is, a polynomial-time algorithm that delivers a solution of value at least  $\rho$  times the optimal value. The constant  $\rho$  is sometimes called the *performance guarantee* of the algorithm. We will also use the term “ $\rho$ -approximation algorithm” for randomized polynomial-time algorithms that deliver solutions whose expected value is at least  $\rho$  times the optimal. Sahni and Gonzales [1976] presented a  $\frac{1}{2}$ -approximation algorithm for the MAX CUT problem. Their algorithm iterates through the vertices and decides whether or not to assign vertex  $i$  to  $S$  based on which placement maximizes the weight of the cut of vertices 1 to  $i$ . This algorithm is essentially equivalent to the randomized algorithm that flips an unbiased coin for each vertex to decide which vertices are assigned to the set  $S$ . Since 1976, a number of researchers have presented approximation algorithms for the unweighted MAX CUT problem with performance guarantees of

$$\frac{1}{2} + \frac{1}{2m} \quad [\text{Vitányi 1981}],$$

$$\frac{1}{2} + \frac{n-1}{4m} \quad [\text{Poljak and Turzík 1982}],$$

$$\frac{1}{2} + \frac{1}{2n} \quad [\text{Haglin and Venkatesan 1991}],$$

and

$$\frac{1}{2} + \frac{1}{2\Delta} \quad [\text{Hofmeister and Lefmann 1995}],$$

(where  $n = |V|$ ,  $m = |E|$  and  $\Delta$  denotes the maximum degree), but no progress was made in improving the constant in the performance guarantee beyond that of Sahni and Gonzales's straightforward algorithm.

We present a simple, randomized  $(\alpha - \epsilon)$ -approximation algorithm for the maximum cut problem where

$$\alpha = \min_{0 \leq \theta \leq \pi} \frac{2}{\pi} \frac{\theta}{1 - \cos \theta} > 0.87856,$$

and  $\epsilon$  is any positive scalar. The algorithm represents the first substantial progress in approximating the MAX CUT problem in nearly twenty years. The algorithm for MAX CUT also leads directly to a randomized  $(\alpha - \epsilon)$ -approximation algorithm for the maximum 2-satisfiability problem (MAX 2SAT). The best previously known algorithm for this problem has a performance guarantee of  $\frac{3}{4}$  and is due to Yannakakis [1994]. A somewhat simpler  $\frac{3}{4}$ -approximation algorithm was given in Goemans and Williamson [1994b]. The improved 2SAT algorithm leads to .7584-approximation algorithm for the overall MAX SAT problem, fractionally better than Yannakakis'  $\frac{3}{4}$ -approximation algorithm for MAX SAT. Finally, a slight extension of our analysis yields a  $(\beta - \epsilon)$ -approximation algorithm for the maximum directed cut problem (MAX DICUT), where

$$\beta = \min_{0 \leq \theta < \arccos(-1/3)} \frac{2}{\pi} \frac{2\pi - 3\theta}{1 + 3\cos \theta} > 0.79607.$$

The best previously known algorithm for MAX DICUT has a performance guarantee of  $\frac{1}{4}$  [Papadimitriou and Yannakakis 1991].

Our algorithm depends on a means of randomly rounding a solution to a nonlinear relaxation of the MAX CUT problem. This relaxation can either be seen as a *semidefinite program* or as an *eigenvalue minimization problem*. To our knowledge, this is the first time that semidefinite programs have been used in the design and analysis of approximation algorithms. The relaxation plays a crucial role in allowing us to obtain a better performance guarantee: previous approximation algorithms compared the value of the solution obtained to the total sum of the weights  $\sum_{i < j} w_{ij}$ . This sum can be arbitrarily close to twice the value of the maximum cut.

A semidefinite program is the problem of optimizing a linear function of a symmetric matrix subject to linear equality constraints and the constraint that the matrix be positive semidefinite. Semidefinite programming is a special case of convex programming and also of the so-called *linear programming over cones* or *cone-LP* since the set of positive semidefinite matrices constitutes a convex cone. To some extent, semidefinite programming is very similar to linear programming; see Alizadeh [1995] for a comparison. It inherits the very elegant duality theory of cone-LP (see Wolkowicz [1981] and the exposition by Alizadeh [1995]). The simplex method can be generalized to semidefinite programs (Pataki [1995]). Given any  $\epsilon > 0$ , semidefinite programs can be solved within an additive error of  $\epsilon$  in polynomial time ( $\epsilon$  is part of the input, so the running time dependence on  $\epsilon$  is polynomial in  $\log 1/\epsilon$ ). This can be done through the ellipsoid algorithm (Grötschel et al. [1988]) and other polynomial-time algorithms for convex programming (Vaidya [1989]), as well as interior-point methods (Nesterov and Nemirovskii [1989; 1994] and Alizadeh [1995]). To terminate in polynomial time, these algorithms implicitly assume some requirement on the feasible space or on the size of the optimum solution; for details,

see Grötschel et al. [1988] and Section 3.3 of Alizadeh [1995]. Since the work of Nesterov and Nemirovskii, and Alizadeh, there has been much development in the design and analysis of interior-point methods for semidefinite programming; for several references available at the time of writing of this paper, see the survey paper by Vandenberghe and Boyd [1996].

Semidefinite programming has many interesting applications in a variety of areas including control theory, nonlinear programming, geometry, and combinatorial optimization.<sup>1</sup> In combinatorial optimization, the importance of semidefinite programming is that it leads to tighter relaxations than the classical linear programming relaxations for many graph and combinatorial problems. A beautiful application of semidefinite programming is the work of Lovász [1979] on the Shannon capacity of a graph. In conjunction with the polynomial-time solvability of semidefinite programs, this leads to the only known polynomial-time algorithm for finding the largest stable set in a perfect graph (Grötschel et al. [1981]). More recently, there has been increased interest in semidefinite programming from a combinatorial point-of-view.<sup>2</sup> This started with the work of Lovász and Schrijver [1989; 1990], who developed a machinery to define tighter and tighter relaxations of any integer program based on quadratic and semidefinite programming. Their papers demonstrated the wide applicability and the power of semidefinite programming for combinatorial optimization problems. Our use of semidefinite programming relaxations was inspired by these papers, and by the paper of Alizadeh [1995].

For MAX CUT, the semidefinite programming relaxation we consider is equivalent to an eigenvalue minimization problem proposed by Delorme and Poljak [1993a; 1993b]. An eigenvalue minimization problem consists of minimizing a decreasing sum of the eigenvalues  $\lambda_i$  of a matrix subject to equality constraints on the matrix; that is, minimizing  $\sum_i m_i \lambda_i$ , where  $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$  and  $m_1 \geq m_2 \geq \dots \geq m_n \geq 0$ . The equivalence of the semidefinite program we consider and the eigenvalue bound of Delorme and Poljak was established by Poljak and Rendl [1995a]. Building on work by Overton and Womersley [1992; 1993], Alizadeh [1995] has shown that eigenvalue minimization problems can in general be formulated as semidefinite programs. This is potentially quite useful, since there is an abundant literature on eigenvalue bounds for combinatorial optimization problems; see the survey paper by Mohar and Poljak [1993].

As shown by Poljak and Rendl [1994; 1995b] and Delorme and Poljak [1993c], the eigenvalue bound provides a very good bound on the maximum cut in practice. Delorme and Poljak [1993a; 1993b] study the worst-case ratio between the maximum cut and their eigenvalue bound. The worst instance they are aware of is the 5-cycle for which the ratio is

$$\frac{32}{25 + 5\sqrt{5}} = 0.88445\dots,$$

but they were unable to prove a bound better than 0.5 in the worst case. Our result implies a worst-case bound of  $\alpha$ , very close to the bound for the 5-cycle.

<sup>1</sup> See, for example, Nesterov and Nemirovskii [1994], Boyd et al. [1994], Vandenberghe and Boyd [1996], and Alizadeh [1995], and the references therein.

<sup>2</sup> See, for example, Lovász and Schrijver [1989; 1990], Alizadeh [1995], Poljak and Rendl [1995a], Feige and Lovász [1992], and Lovász [1992].

The above discussion on the worst-case behavior indicates that straightforward modifications of our technique will not lead to significant improvements in the MAX CUT result. Furthermore, MAX CUT, MAX 2SAT, and MAX DICUT are MAX SNP-hard [1991], and so it is known that there exists a constant  $c < 1$  such that a  $c$ -approximation algorithm for any of these problems would imply that  $P = NP$  [Arora et al. 1992]. Bellare et al. [unpublished manuscript] have shown that  $c$  is as small as  $83/84$  for MAX CUT and  $95/96$  for MAX 2SAT. Since bidirected instances of MAX DICUT are equivalent to instances of MAX CUT, the bound for MAX CUT also applies to MAX DICUT.

Since the appearance of an abstract of this paper, Feige and Goemans [1995] have extended our technique to yield a .931-approximation algorithm for MAX 2SAT and a .859-approximation algorithm for MAX DICUT. By using semidefinite programming and similar rounding ideas, Karger et al. [1994] have been able to show how to color a  $k$ -colorable graph with  $\tilde{O}(n^{1-(3/(k+1))})$  colors in polynomial time. Frieze and Jerrum [1996] have used the technique to devise approximation algorithms for the maximum  $k$ -way cut problem that improve on the best previously known  $1 - 1/k$  performance guarantee. Chor and Sudan [1995] apply ideas from this paper to the “betweenness” problem. Thus, it seems likely that the techniques in this paper will continue to prove useful in designing approximation algorithms.

We expect that in practice the performance of our algorithms will be much better than the worst-case bounds. We have performed some preliminary computational experiments with the MAX CUT algorithm which show that on a number of different types of random graphs the algorithm is usually within .96 of the optimal solution.

A preliminary version of this paper [Goemans and Williamson 1994a] presented a method to obtain deterministic versions of our approximation algorithm with the same performance guarantees. However, the method given had a subtle error, as was pointed out to us by several researchers. Mahajan and Ramesh [1995] document the error and propose their own derandomization scheme for our algorithms.

The paper is structured as follows: We present the randomized algorithm for MAX CUT in Section 2, and its analysis in Section 3. We elaborate on our semidefinite programming bound and its relationship with other work on the MAX CUT problem in Section 4. The quality of the relaxation is investigated in Section 5, and computational results are presented in Section 6. In Section 7, we show how to extend the algorithm to an algorithm for MAX 2SAT, MAX SAT, MAX DICUT, and other problems. We conclude with a few remarks and open problems in Section 8.

## 2. The Randomized Approximation Algorithm for MAX CUT

Given a graph with vertex set  $V = \{1, \dots, n\}$  and nonnegative weights  $w_{ij} = w_{ji}$  for each pair of vertices  $i$  and  $j$ , the weight of the maximum cut  $w(S, \bar{S})$  is given by the following integer quadratic program:

$$\begin{aligned}
 & \text{Maximize} && \frac{1}{2} \sum_{i < j} w_{ij} (1 - y_i y_j) \\
 (Q) \quad & \text{subject to:} && y_i \in \{-1, 1\} \quad \forall i \in V.
 \end{aligned}$$

To see this, note that the set  $S = \{i | y_i = 1\}$  corresponds to a cut of weight  $w(S, \bar{S}) = \frac{1}{2} \sum_{i < j} w_{ij}(1 - y_i y_j)$ .

Since solving this integer quadratic program is NP-complete, we consider relaxations of  $(Q)$ . Relaxations of  $(Q)$  are obtained by relaxing some of the constraints of  $(Q)$ , and extending the objective function to the larger space; thus, all possible solutions of  $(Q)$  are feasible for the relaxation, and the optimal value of the relaxation is an upper bound on the optimal value of  $(Q)$ . We can interpret  $(Q)$  as restricting  $y_i$  to be a 1-dimensional vector of unit norm. Some very interesting relaxations can be defined by allowing  $y_i$  to be a multidimensional vector  $v_i$  of unit Euclidean norm. Since the linear space spanned by the vectors  $v_i$  has dimension at most  $n$ , we can assume that these vectors belong to  $\mathbb{R}^n$  (or  $\mathbb{R}^m$  for some  $m \leq n$ ), or more precisely to the  $n$ -dimensional unit sphere  $S_n$  (or  $S_m$  for  $m \leq n$ ). To ensure that the resulting optimization problem is indeed a relaxation, we need to define the objective function in such a way that it reduces to  $\frac{1}{2} \sum_{i < j} w_{ij}(1 - y_i y_j)$  in the case of vectors lying in a 1-dimensional space. There are several natural ways of guaranteeing this property. For example, one can replace  $(1 - y_i y_j)$  by  $(1 - v_i \cdot v_j)$ , where  $v_i \cdot v_j$  represents the inner product (or dot product) of  $v_i$  and  $v_j$ . The resulting relaxation is denoted by  $(P)$ :

$$\begin{aligned} & \text{Maximize} \quad \frac{1}{2} \sum_{i < j} w_{ij}(1 - v_i \cdot v_j) \\ (P) \quad & \text{subject to: } v_i \in S_n \quad \forall i \in V. \end{aligned}$$

We will show in Section 4 that we can solve this relaxation using semidefinite programming. We can now present our simple randomized algorithm for the MAX CUT problem.

- (1) Solve  $(P)$ , obtaining an optimal set of vectors  $v_i$ .
- (2) Let  $r$  be a vector uniformly distributed on the unit sphere  $S_n$ .
- (3) Set  $S = \{i | v_i \cdot r \geq 0\}$ .

In other words, we choose a random hyperplane through the origin (with  $r$  as its normal) and partition the vertices into those vectors that lie “above” the plane (i.e., have a nonnegative inner product with  $r$ ) and those that lie “below” it (i.e., have a negative inner product with  $r$ ). The motivation for this randomized step comes from the fact that  $(P)$  is independent of the coordinate system: applying any orthonormal transformation to a set of vectors results in a solution with the same objective value.

Let  $W$  denote the value of the cut produced in this way, and  $E[W]$  its expectation. We will show in Theorem 3.1 in Section 3 that, given any set of vectors  $v_i \in S_n$ , the expected weight of the cut defined by a random hyperplane is

$$E[W] = \sum_{i < j} w_{ij} \frac{\arccos(v_i \cdot v_j)}{\pi}.$$

We will also show in Theorem 3.3 that

$$E[W] \geq \alpha \cdot \frac{1}{2} \sum_{i < j} w_{ij}(1 - v_i \cdot v_j),$$

where

$$\alpha = \min_{0 \leq \theta \leq \pi} \frac{2}{\pi} \frac{\theta}{1 - \cos \theta} > .878.$$

If  $Z_{MC}^*$  is the optimal value of the maximum cut and  $Z_P^*$  is the optimal value of the relaxation (P), then since the expected weight of the cut generated by the algorithm is equal to  $E[W] \geq \alpha Z_P^* \geq \alpha Z_{MC}^*$ , the algorithm has a performance guarantee of  $\alpha$  for the MAX CUT problem.

We must argue that the algorithm can be implemented in polynomial time. We assume that the weights are integral. In Section 4, we show that the program (P) is equivalent to a semidefinite program. Then we will show that, by using an algorithm for semidefinite programming, we can obtain, for any  $\epsilon > 0$ , a set of vectors  $v_i$ 's of value greater than  $Z_P^* - \epsilon$  in time polynomial in the input size and  $\log 1/\epsilon$ . On these approximately optimal vectors, the randomized algorithm will produce a cut of expected value greater than or equal to  $\alpha(Z_P^* - \epsilon) \geq (\alpha - \epsilon)Z_{MC}^*$ . The point on the unit sphere  $S_n$  can be generated by drawing  $n$  values  $x_1, x_2, \dots, x_n$  independently from the standard normal distribution, and normalizing the vector obtained (see Knuth [1981, p. 130]); for our purposes, there is no need to normalize the resulting vector  $x$ . The standard normal distribution can be simulated using the uniform distribution between 0 and 1 (see Knuth [1981, p. 117]). The algorithm is thus a randomized  $(\alpha - \epsilon)$ -approximation algorithm for MAX CUT.

### 3. Analysis of the Algorithm

In this section, we analyze the performance of the algorithm. We first analyze the general case and then consider the case in which the maximum cut is large and the generalization to negative edge weights. We conclude this section with a new formulation for the MAX CUT problem.

Let  $\{v_1, \dots, v_n\}$  be any vectors belonging to  $S_n$ , and let  $E[W]$  be the expected value of the cut  $w(S, \bar{S})$  produced by the randomized algorithm given in the previous section. We start by characterizing the expected value of the cut.

THEOREM 3.1

$$E[W] = \frac{1}{\pi} \sum_{i < j} w_{ij} \arccos(v_i \cdot v_j).$$

Given a vector  $r$  drawn uniformly from the unit sphere  $S_n$ , we know by the linearity of expectation that

$$E[W] = \sum_{i < j} w_{ij} \cdot \Pr[\text{sgn}(v_i \cdot r) \neq \text{sgn}(v_j \cdot r)],$$

where  $\text{sgn}(x) = 1$  if  $x \geq 0$ , and  $-1$  otherwise. Theorem 3.1 is thus implied by the following lemma.

LEMMA 3.2

$$\Pr[\text{sgn}(v_i \cdot r) \neq \text{sgn}(v_j \cdot r)] = \frac{1}{\pi} \arccos(v_i \cdot v_j).$$

PROOF. A restatement of the lemma is that the probability the random hyperplane separates the two vectors is directly proportional to the angle between the two vectors; that is, it is proportional to the angle  $\theta = \arccos(v_i \cdot v_j)$ . By symmetry,  $\Pr[\text{sgn}(v_i \cdot r) \neq \text{sgn}(v_j \cdot r)] = 2 \Pr[v_i \cdot r \geq 0, v_j \cdot r < 0]$ . The set  $\{r : v_i \cdot r \geq 0, v_j \cdot r < 0\}$  corresponds to the intersection of two half-spaces whose dihedral angle is precisely  $\theta$ ; its intersection with the sphere is a spherical digon of angle  $\theta$  and, by symmetry of the sphere, thus has measure equal to  $\theta/2\pi$  times the measure of the full sphere. In other words,  $\Pr[v_i \cdot r \geq 0, v_j \cdot r < 0] = \theta/2\pi$ , and the lemma follows.  $\square$

Our main theorem is the following: As we have argued above, this theorem applied to vectors  $v_i$ 's of value greater than  $Z_p^* - \epsilon$  implies that the algorithm has a performance guarantee of  $\alpha - \epsilon$ . Recall that we defined

$$\alpha = \min_{0 \leq \theta \leq \pi} \frac{2}{\pi} \frac{\theta}{1 - \cos \theta}.$$

THEOREM 3.3

$$E[W] \geq \alpha \frac{1}{2} \sum_{i < j} w_{ij} (1 - v_i \cdot v_j).$$

By using Theorem 3.1 and the nonnegativity of  $w_{ij}$ , the result follows from the following lemma applied to  $y = v_i \cdot v_j$ .

LEMMA 3.4. For  $-1 \leq y \leq 1$ ,  $1 \arccos(y)/\pi \geq \alpha \cdot \frac{1}{2}(1 - y)$ .

PROOF. The expression for  $\alpha$  follows straightforwardly by using the change of variables  $\cos \theta = y$ . See Figure 1, part (i).  $\square$

The quality of the performance guarantee is established by the following lemma.

LEMMA 3.5.  $\alpha > .87856$ .

PROOF. Using simple calculus, one can see that  $\alpha$  achieves its value for  $\theta = 2.331122\dots$ , the nonzero root of  $\cos \theta + \theta \sin \theta = 1$ . To formally prove the bound of 0.87856, we first observe that

$$\frac{2}{\pi} \frac{\theta}{1 - \cos \theta} \geq 1$$

for  $0 < \theta \leq \pi/2$ . The concavity of  $f(\theta) = 1 - \cos \theta$  in the range  $\pi/2 \leq \theta \leq \pi$  implies that, for any  $\theta_0$ , we have  $f(\theta) \leq f(\theta_0) + (\theta - \theta_0)f'(\theta)$ , or  $1 - \cos \theta \leq 1 - \cos \theta_0 + (\theta - \theta_0)\sin \theta_0 = \theta \sin \theta_0 + (1 - \cos \theta_0 - \theta_0 \sin \theta_0)$ . Choosing  $\theta_0 = 2.331122$  for which  $1 - \cos \theta_0 - \theta_0 \sin \theta_0 < 0$ , we derive that  $1 - \cos \theta < \theta \sin \theta_0$ , implying that

$$\alpha > \frac{2}{\pi \sin \theta_0} > 0.87856. \quad \square$$

We have analyzed the expected value of the cut returned by the algorithm. For randomized algorithms, one can often also give high probability results. In this case, however, even computing the variance of the cut value analytically is difficult. Nevertheless, one could use the fact that  $W$  is bounded (by  $\sum_{i < j} w_{ij}$ ) to give lower bounds on the probability that  $W$  is less than  $(1 - \epsilon)E[W]$ .



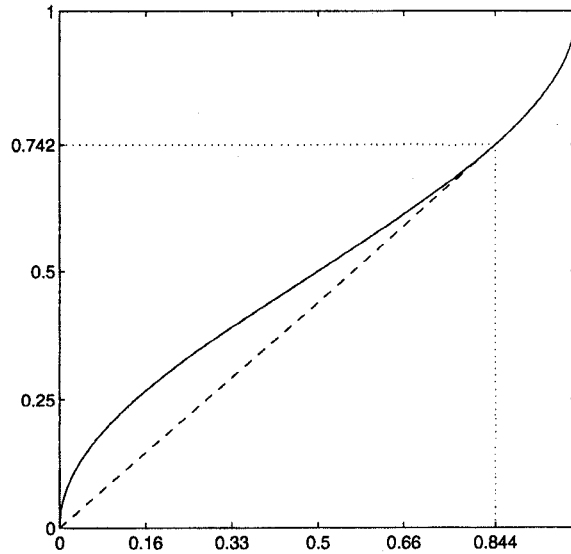


FIG. 1. (i) Plot of  $z = \arccos(y)/\pi$  as a function of  $t = \frac{1}{2}(1 - y)$ . The ratio  $z/t$  is thus the slope of the line between  $(0,0)$  and  $(z,t)$ . The minimum slope is precisely  $\alpha = 0.742/0.844$  and corresponds to the dashed line. (ii) The plot also represents  $h(t) = \arccos(1 - 2t)/\pi$  as a function of  $t$ . As  $t$  approaches 1,  $h(t)/t$  also approaches 1. (iii) The dashed line corresponds to  $\tilde{h}$  between 0 and  $\gamma \approx .844$ .

**3.1 ANALYSIS WHEN THE MAXIMUM CUT IS LARGE.** We can refine the analysis and prove a better performance guarantee whenever the value of the relaxation  $(P)$  constitutes a large fraction of the total weight. Define  $W_{\text{tot}} = \sum_{i < j} w_{ij}$  and  $h(t) = \arccos(1 - 2t)/\pi$ . Let  $\gamma$  be the value of  $t$  attaining the minimum of  $h(t)/t$  in the interval  $(0, 1]$ . The value of  $\gamma$  is approximately .84458.

**THEOREM 3.1.1.** *Let*

$$A = \frac{1}{W_{\text{tot}}} \sum_{i < j} w_{ij} \frac{1 - v_i \cdot v_j}{2}.$$

*If  $A \geq \gamma$ , then*

$$E[W] \geq \frac{h(A)}{A} \sum_{i < j} w_{ij} \frac{1 - v_i \cdot v_j}{2}.$$

The theorem implies a performance guarantee of  $h(A)/A - \epsilon$  when  $A \geq \gamma$ . As  $A$  varies between  $\gamma$  and 1, one easily verifies that  $h(A)/A$  varies between  $\alpha$  and 1 (see Figure 1, part (ii)).

**PROOF.** Letting  $\lambda_e = w_{ij}/W_{\text{tot}}$  and  $x_e = (1 - v_i \cdot v_j)/2$  for  $e = (i, j)$ , we can rewrite  $A$  as  $A = \sum_e \lambda_e x_e$ . The expected weight of the cut produced by

the algorithm is equal to

$$\begin{aligned} E[W] &= \sum_{i < j} w_{ij} \frac{\arccos(v_i \cdot v_j)}{\pi} = W_{\text{tot}} \sum_e \lambda_e \frac{\arccos(1 - 2x_e)}{\pi} \\ &= W_{\text{tot}} \sum_e \lambda_e h(x_e). \end{aligned}$$

To bound  $E[W]$ , we evaluate

$$\begin{aligned} \text{Min} \quad & \sum_e \lambda_e h(x_e) \\ \text{subject to: } & \sum_e \lambda_e x_e = A \\ & 0 \leq x_e \leq 1. \end{aligned}$$

Consider the relaxation obtained by replacing  $h(t)$  by the largest (pointwise) convex function  $\tilde{h}(t)$  smaller or equal to  $h(t)$ . It is easy to see that  $\tilde{h}(t)$  is linear with slope  $\alpha$  between 0 and  $\gamma$ , and then equal to  $h(t)$  for any  $t$  greater than  $\gamma$ . See Figure 1, part (iii). But for  $A \geq \gamma$ ,

$$\sum_e \lambda_e h(x_e) \geq \sum_e \lambda_e \tilde{h}(x_e) \geq \tilde{h}\left(\sum_e \lambda_e x_e\right) = \tilde{h}(A) = h(A),$$

where we have used the fact that  $\sum_e \lambda_e = 1$ ,  $\lambda_e \geq 0$  and that  $\tilde{h}$  is a convex function. This shows that

$$E[W] \geq W_{\text{tot}} h(A) = \frac{h(A)}{A} \sum_{i < j} w_{ij} \frac{1 - v_i \cdot v_j}{2},$$

proving the result.  $\square$

**3.2 ANALYSIS FOR NEGATIVE EDGE WEIGHTS.** So far, we have assumed that the weights are nonnegative. In several practical problems, some edge weights are negative [Barahona et al. 1988]. In this case the definition of the performance guarantee has to be modified since the optimum value could be positive or negative. We now give a corresponding generalization of Theorem 3.3 to arbitrary weights.

**THEOREM 3.2.1.** *Let  $W_- = \sum_{i < j} w_{ij}^-$ , where  $x^- = \min(0, x)$ . Then*

$$\{E[W] - W_-\} \geq \alpha \left\{ \frac{1}{2} \sum_{i < j} w_{ij} (1 - v_i \cdot v_j) - W_- \right\}.$$

**PROOF.** The quantity  $E[W] - W_-$  can be written as

$$\sum_{i < j: w_{ij} > 0} w_{ij} \frac{\arccos(v_i \cdot v_j)}{\pi} + \sum_{i < j: w_{ij} < 0} |w_{ij}| \left( 1 - \frac{\arccos(v_i \cdot v_j)}{\pi} \right).$$

Similarly,  $\left\{ \frac{1}{2} \sum_{i < j} w_{ij} (1 - v_i \cdot v_j) - W_- \right\}$  is equal to

$$\sum_{i < j: w_{ij} > 0} w_{ij} \frac{1 - v_i \cdot v_j}{2} + \sum_{i < j: w_{ij} < 0} |w_{ij}| \frac{1 + v_i \cdot v_j}{2}.$$

The result therefore follows from Lemma 3.4 and the following variation of it.  $\square$

LEMMA 3.2.2. For  $-1 \leq z \leq 1$ ,  $1 - \arccos(z)/\pi \geq \alpha \cdot \frac{1}{2}(1 + z)$ .

PROOF. The lemma follows from Lemma 3.4 by using a change of variables  $z = -y$  and noting that  $\pi - \arccos(z) = \arccos(-z)$ .  $\square$

3.3 A NEW FORMULATION OF MAX CUT. An interesting consequence of our analysis is a new nonlinear formulation of the maximum cut problem. Consider the following nonlinear program:

$$\begin{aligned} & \text{Maximize} && \sum_{i < j} w_{ij} \frac{\arccos(v_i \cdot v_j)}{\pi} \\ (R) \quad & \text{subject to:} && v_i \in S_n \quad \forall i \in V. \end{aligned}$$

Let  $Z_R^*$  denote the optimal value of this program.

THEOREM 3.3.1.  $Z_R^* = Z_{MC}^*$ .

PROOF. We first show that  $Z_R^* \geq Z_{MC}^*$ . This follows since (R) is a relaxation of (Q): the objective function of (R) reduces to  $\frac{1}{2} \sum_{i < j} w_{ij} (1 - v_i \cdot v_j)$  in the case of vectors  $v_i$  lying in a 1-dimensional space.

To see that  $Z_R^* \leq Z_{MC}^*$ , let the vectors  $v_i$  denote the optimal solution to (R). From Theorem 3.1, we see that the randomized algorithm gives a cut whose expected value is exactly  $Z_R^*$ , implying that there must exist a cut of value at least  $Z_R^*$ .  $\square$

#### 4. Relaxations and Duality

In this section, we address the question of solving the relaxation (P). We do so by showing that (P) is equivalent to a semidefinite program. We then explore the dual of this semidefinite program and relate it to the eigenvalue minimization bound of Delorme and Poljak [1993a; 1993b].

4.1 SOLVING THE RELAXATION. We begin by defining some terms and notation. All matrices under consideration are defined over the reals. An  $n \times n$  matrix  $A$  is said to be positive semidefinite if for every vector  $x \in \mathbb{R}^n$ ,  $x^T A x \geq 0$ . The following statements are equivalent for a symmetric matrix  $A$  (see, e.g., Lancaster and Tismenetsky [1985]); (i)  $A$  is positive semidefinite, (ii) all eigenvalues of  $A$  are nonnegative, and (iii) there exists a matrix  $B$  such that  $A = B^T B$ . In (iii),  $B$  can either be a (possibly singular)  $n \times n$  matrix, or an  $m \times n$  matrix for some  $m \leq n$ . Given a symmetric positive semidefinite matrix  $A$ , an  $m \times n$  matrix  $B$  of full row-rank satisfying (iii) can be obtained in  $O(n^3)$  time using an incomplete Cholesky decomposition [Golub and Van Loan 1983, p. 90, P5.2-3].

Using the decomposition  $Y = B^T B$ , one can see that a positive semidefinite  $Y$  with  $y_{ii} = 1$  corresponds precisely to a set of unit vectors  $v_1, \dots, v_n \in S_m$ : simply correspond the vector  $v_i$  to the  $i$ th column of  $B$ . Then  $y_{ij} = v_i \cdot v_j$ . The matrix  $Y$  is known as the Gram matrix of  $\{v_1, \dots, v_n\}$  [Lancaster and Tismenetsky 1985, p. 110]. Using this equivalence, we can reformulate (P) as a semidefinite program:

$$\begin{aligned} Z_P^* = \text{Max} \quad & \frac{1}{2} \sum_{i < j} w_{ij} (1 - y_{ij}) \\ (SD) \quad & \\ \text{subject to: } & y_{ii} = 1 \quad \forall i \in V, \\ & Y \text{ symmetric positive semidefinite} \end{aligned}$$

where  $Y = (y_{ij})$ . The feasible solutions to (SD) are often referred to as *correlation matrices* [Grone et al. 1990]. Strictly speaking, we cannot solve (SD) to optimality in polynomial time; the optimal value  $Z_P^*$  might in fact be irrational. However, using an algorithm for semidefinite programming, one can obtain, for any  $\epsilon > 0$ , a solution of value greater than  $Z_P^* - \epsilon$  in time polynomial in the input size and  $\log 1/\epsilon$ . For example, Alizadeh's adaptation of Ye's interior-point algorithm to semidefinite programming [Alizadeh 1995] performs  $O(\sqrt{n}(\log W_{\text{tot}} + \log 1/\epsilon))$  iterations. By exploiting the simple structure of the problem (SD) as is indicated in Rendl et al. [1993] (see also Vandenberghe and Boyd [1996, Sect. 7.4]), each iteration can be implemented in  $O(n^3)$  time. Once an almost optimal solution to (SD) is found, one can use an incomplete Cholesky decomposition to obtain vectors  $v_1, \dots, v_n \in S_m$  for some  $m \leq n$  such that

$$\frac{1}{2} \sum_{i < j} w_{ij} (1 - v_i \cdot v_j) \geq Z_P^* - \epsilon.$$

Among all optimum solutions to (SD), one can show the existence of a solution of low rank. Grone et al. [1990] show that any extreme solution of (SD) (i.e., which cannot be expressed as the strict convex combination of other feasible solutions) has rank at most  $l$  where

$$\frac{l(l+1)}{2} \leq n,$$

that is,

$$l \leq \frac{\sqrt{8n+1} - 1}{2} < \sqrt{2n}.$$

For related results, see Li and Tam [1994], Christensen and Vesterstrøm [1979], Loewy [1980], and Laurent and Poljak [1996]. This means that there exists a primal optimum solution  $Y^*$  to (SD) of rank less than  $\sqrt{2n}$ , and that the optimum vectors  $v_i$  of (P) can be embedded in  $\mathbb{R}^m$  with  $m < \sqrt{2n}$ . This result also follows from a more general statement about semidefinite programs due to Barvinok [1995] and implicit in Pataki [1994]: any extreme solution of a semidefinite program with  $k$  linear equalities has rank at most  $l$  where  $l(l+1)/2 \leq k$ .

**4.2 THE SEMIDEFINITE DUAL.** As mentioned in the introduction, there is an elegant duality theory for semidefinite programming. We now turn to discussing the dual of the program (SD). It is typical to assume that the objective function of a semidefinite program is symmetric. For this purpose, we can rewrite the objective function of (SD) as  $\frac{1}{4} \sum_{i=1}^n \sum_{j=1}^n w_{ij}(1 - y_{ij})$ , or even as  $\frac{1}{2}W_{tot} - \frac{1}{4} \sum_i \sum_j w_{ij}y_{ij}$ . In matrix form, the objective function can be conveniently written as  $\frac{1}{2}W_{tot} - \frac{1}{4}Tr(WY)$ , where  $W = (w_{ij})$  and  $Tr$  denotes the trace.

The dual of (SD) has a very simple description:

$$Z_D^* = \frac{1}{2}W_{tot} + \frac{1}{4} \text{Min} \sum_i \gamma_i$$

(D) subject to:  $W + \text{diag}(\gamma)$  positive semidefinite,

where  $\text{diag}(\gamma)$  denotes the diagonal matrix whose  $i$ th diagonal entry is  $\gamma_i$ . The dual has a simple interpretation. Since  $W + \text{diag}(\gamma)$  is positive semidefinite, it can be expressed as  $C^TC$ ; in other words, the weight  $w_{ij}$  can be viewed as  $c_i \cdot c_j$  for some vectors  $c_i$ 's and  $\gamma_i = c_i \cdot c_i = \|c_i\|^2$ . The weight of any cut is thus  $w(S, \bar{S}) = (\sum_{i \in S} c_i) \cdot (\sum_{j \notin S} c_j)$ , which is never greater than

$$\left\| \frac{\sum_{i \in V} c_i}{2} \right\|^2 = \frac{1}{2}W_{tot} + \frac{1}{4} \sum_{i \in V} \gamma_i.$$

Showing weak duality between (P) and (D), namely that  $Z_P^* \leq Z_D^*$ , is easy. Consider any primal feasible solution  $Y$  and any dual vector  $\gamma$ . Since both  $Y$  and  $W + \text{diag}(\gamma)$  are positive semidefinite, we derive that  $Tr((\text{diag}(\gamma) + W)Y) \geq 0$  (see Lancaster and Tismenetsky [1985, p. 218, ex. 14]). But  $Tr((\text{diag}(\gamma) + W)Y) = Tr(\text{diag}(\gamma)Y) + Tr(WY) = \sum_i \gamma_i + Tr(WY)$ , implying that the difference of the dual objective function value and the primal objective function value is nonnegative.

For semidefinite programs in their full generality, there is no guarantee that the primal optimum value is equal to the dual optimum value. Also, the maximum (respectively, minimum) is in fact a supremum (respectively, infimum) and there is no guarantee that the supremum (respectively, infimum) is attained. These, however, are pathological cases. Our programs (SD) and (D) behave nicely; both programs attain their optimum values, and these values are equal (i.e.,  $Z_P^* = Z_D^*$ ). This can be shown in a variety of ways (see Poljak and Rendl [1995a], Alizadeh [1995], and Barvinok [1995]).

Given that strong duality holds in our case, the argument showing weak duality implies that, for the optimum primal solution  $Y^*$  and the optimum dual solution  $\gamma^*$ , we have  $Tr((\text{diag}(\gamma^*) + W)Y^*) = 0$ . Since both  $\text{diag}(\gamma^*) + W$  and  $Y^*$  are positive semidefinite, we derive that  $(\text{diag}(\gamma^*) + W)Y^* = 0$  (see Lancaster and Tismenetsky [1985, p. 218, ex. 14]). This is the strong form of complementary slackness for semidefinite programs (see Alizadeh [1995]); the component-wise product expressed by the trace is replaced by matrix multiplication. This implies, for example, that  $Y^*$  and  $\text{diag}(\gamma^*) + W$  share a system of eigenvectors and that  $\text{rank}(Y^*) + \text{rank}(\text{diag}(\gamma^*) + W) \leq n$ .

**4.3 THE EIGENVALUE BOUND OF DELORME AND POLJAK.** The relaxation (D) (and thus (P)) is also equivalent to an eigenvalue upper bound on the value of the maximum cut  $Z_{MC}^*$  introduced by Delorme and Poljak [1993a; 1993b]. To describe the bound, we first introduce some notation. The Lapla-

cian matrix  $L = (l_{ij})$  is defined by  $l_{ij} = -w_{ij}$  for  $i \neq j$  and  $l_{ii} = \sum_{k=1}^n w_{ik}$ . The maximum eigenvalue of a matrix  $A$  is denoted by  $\lambda_{\max}(A)$ .

LEMMA 4.3.1 [DELORME AND POLJAK [1993b].] *Let  $u \in \mathbb{R}^n$  satisfy  $u_1 + \dots + u_n = 0$ . Then*

$$\frac{n}{4} \lambda_{\max}(L + \text{diag}(u))$$

*is an upper bound on  $Z_{MC}^*$ .*

PROOF. The proof is simple. Let  $y$  be an optimal solution to the integer quadratic program  $(Q)$ . Notice that  $y^T L y = 4Z_{MC}^*$ . By using the Rayleigh principle ( $\lambda_{\max}(M) = \max_{\|x\|=1} x^T M x$ ), we obtain

$$\begin{aligned} \lambda_{\max}(L + \text{diag}(u)) &\geq \frac{y^T (L + \text{diag}(u)) y}{y^T y} \\ &= \frac{1}{n} \left( y^T L y + \sum_{i=1}^n y_i^2 u_i \right) \\ &= \frac{4Z_{MC}^*}{n}, \end{aligned}$$

proving the lemma.  $\square$

A vector  $u$  satisfying  $\sum_{i=1}^n u_i = 0$  is called a *correcting vector*. Let  $g(u) = (n/4) \lambda_{\max}(L + \text{diag}(u))$ . The bound proposed by Delorme and Poljak [1993b] is to optimize  $g(u)$  over all correcting vectors:

$$\begin{aligned} Z_{EIG}^* &= \inf g(u) \\ (EIG) \quad &\text{subject to: } \sum_{i=1}^n u_i = 0. \end{aligned}$$

As mentioned in the introduction, eigenvalue minimization problems can be formulated as semidefinite programs. For MAX CUT, the equivalence between  $(SD)$  and  $(EIG)$  was established by Poljak and Rendl [1995a].

For completeness, we derive the equivalence between  $(EIG)$  and the dual  $(D)$ . For the optimum dual vector  $\gamma^*$ , the smallest eigenvalue of  $\text{diag}(\gamma^*) + W$  must be 0, since otherwise we could decrease all  $\gamma_i^*$  by some  $\epsilon > 0$  and thus reduce the dual objective function. This can be rewritten as  $\lambda_{\max}(-W - \text{diag}(\gamma^*)) = 0$ . Define  $\lambda$  as  $(\sum_i \gamma_i^* + 2W_{tot})/n$ . By definition,  $Z_D^* = n\lambda/4$ . Moreover, defining  $u_i = \lambda - \gamma_i^* - \sum_j w_{ij}$ , one easily verifies that  $\sum_i u_i = 0$  and that  $-W - \text{diag}(\gamma^*) = L + \text{diag}(u) - \lambda I$ , implying that  $\lambda_{\max}(L + \text{diag}(u)) = \lambda$ . This shows that  $Z_{EIG}^* \leq Z_D^*$ . The converse inequality follows by reversing the argument.

### 5. Quality of the Relaxation

In this section we consider the tightness of our analysis and the quality of the semidefinite bound  $Z_P^*$ . Observe that Theorem 3.3 implies the following corollary:

COROLLARY 5.1. *For any instance of MAX CUT,*

$$\frac{Z_{MC}^*}{Z_P^*} \geq \alpha.$$

For the 5-cycle, Delorme and Poljak [1993b] have shown that

$$\frac{Z_{MC}^*}{Z_{EIG}^*} = \frac{32}{25 + 5\sqrt{5}} = 0.88445\dots,$$

implying that our worst-case analysis is almost tight. One can obtain this bound from the relaxation  $(P)$  by observing that for the 5-cycle 1–2–3–4–5–1, the optimal vectors lie in a 2-dimensional subspace and can be expressed as

$$v_i = \left( \cos\left(\frac{4i\pi}{5}\right), \sin\left(\frac{4i\pi}{5}\right) \right)$$

for  $i = 1, \dots, 5$  corresponding to

$$Z_P^* = \frac{5}{2} \left( 1 + \cos\frac{\pi}{5} \right) = \frac{25 + 5\sqrt{5}}{8}.$$

Since  $Z_{MC}^* = 4$  for the 5-cycle, this yields the bound of Delorme and Poljak. Delorme and Poljak have shown that

$$\frac{Z_{MC}^*}{Z_{EIG}^*} \geq \frac{32}{25 + 5\sqrt{5}}$$

holds for special subclasses of graphs, such as planar graphs or line graphs. However, they were unable to prove a bound better than 0.5 in the absolute worst-case.

Although the worst-case value of  $Z_{MC}^*/Z_P^*$  is not completely settled, there exist instances for which  $E[W]/Z_P^*$  is very close to  $\alpha$ , showing that the analysis of our algorithm is practically tight. Leslie Hall (personal communication) has observed that  $E[W]/Z_P^* \approx .8787$  for the Petersen graph [Bondy and Murty 1976, p. 55]. In Figure 2, we give an unweighted instance for which the ratio is less than .8796 in which the vectors have a nice three-dimensional representation. We have also constructed a weighted instance on 103 vertices for which the ratio is less than .8786. These two instances are based on strongly self-dual polytopes due to Lovász [1983]. A polytope  $P$  in  $\mathbb{R}^n$  is said to be *strongly self-dual* [Lovász 1983], if (i)  $P$  is inscribed in the unit sphere, (ii)  $P$  is circumscribed around the sphere with origin as center and with radius  $r$  for some  $0 < r < 1$ , and (iii) there is a bijection  $\sigma$  between vertices and facets of  $P$  such that, for every vertex  $v$  of  $P$ , the facet  $\sigma(v)$  is orthogonal to the vector  $v$ . For example, in  $\mathbb{R}^2$ , the strongly self-dual polytopes are precisely the regular odd polygons. One can associate a graph  $G = (V, E)$  to a self-dual polytope  $P$ : the vertex set  $V$  corresponds to the vertices of  $P$  and there is an edge  $(v, w)$  if  $w$  belongs to the facet  $\sigma(v)$  (or, equivalently,  $v$  belongs to  $\sigma(w)$ ). For the regular odd polygons, these graphs are simply the odd cycles. Because of conditions (ii) and (iii), the inner product  $v \cdot w$  for any pair of adjacent vertices is equal to  $-r$ . As a result, a strongly self-dual polytope leads to a feasible solution of  $(P)$  of value  $((1 + r)/2)W_{tot}$ . Lovász [1983] gives a recursive construction of a class of strongly self-dual polytopes. One can show that, by choosing the dimension  $n$  large enough, his construction leads to strongly self-dual polytopes for which  $r$  is arbitrarily close to the critical value giving a bound of  $\alpha$ . However, it is unclear whether, in general, for such polytopes, nonnegative weights can be selected such that the vectors given by the polytope

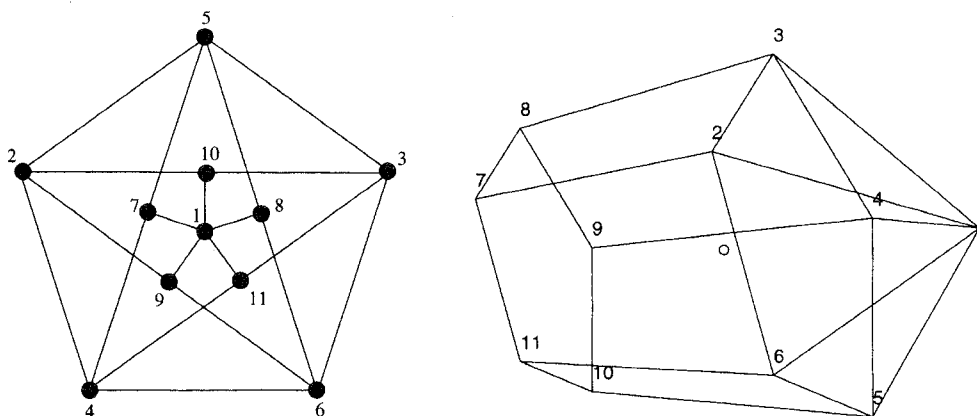


FIG. 2. Graph on 11 vertices for which the ratio  $E[W]/Z_P^*$  is less than .8796 in the unweighted case. The convex hull of the optimum vectors is depicted on the right; the circle represents the center of the sphere.

constitute an *optimum* solution to  $(P)$ . Nevertheless, we conjecture that such instances lead to a proof that  $E[W]/Z_P^*$  can be arbitrarily close to  $\alpha$ . Even if this could be shown, this would not imply anything for the ratio  $Z_{MC}^*/Z_P^*$ .

## 6. Computational Results

In practice, we expect that the algorithm will perform much better than the worst-case bound of  $\alpha$ . Poljak and Rendl [1994; 1995b] (see also Delorme and Poljak [1993c]) report computational results showing that the bound  $Z_{EIG}^*$  is typically less than 2–5% and, in the instances they tried, never worse than 8% away from  $Z_{MC}^*$ . We also performed our own computational experiments, in which the cuts computed by the algorithm were usually within 4% of the semidefinite bound  $Z_P^*$ , and never less than 9% from the bound. To implement the algorithm, we used code supplied by Vanderbei [Rendl et al. 1993] for a special class of semidefinite programs. We used Vanderbei's code to solve the semidefinite program, then we generated 50 random vectors  $r$ . We output the best of the 50 cuts induced. We applied our code to a small subset of the instances considered by Poljak and Rendl [1995b]. In particular, we considered several different types of random graphs, as well as complete geometric graphs defined by Traveling Salesman Problem (TSP) instances from the TSPLIB (see Reinelt [1991]).

For four different types of random graphs, we ran 50 instances on graphs of 50 vertices, 20 on graphs of size 100, and 5 on graphs of size 200. In the Type A random graph, each edge  $(i, j)$  is included with probability  $1/2$ . In the Type B random graph, each edge is given a weight drawn uniformly from the interval  $[-50, 50]$ ; the ratio of Theorem 3.2.1 is used in reporting nearness to the semidefinite bound. In the Type C random graph of size  $n \geq 10$ , an edge  $(i, j)$  is included with probability  $10/n$ , leading to constant expected degree. Finally, in the Type D random graphs, an edge  $(i, j)$  is included with probability .1 if  $i \leq n/2$  and  $j > n/2$  and probability .05 otherwise, leading to a large cut between the vertices in  $[1, \dots, n/2]$  and those in  $[n/2 + 1, \dots, n]$ . We summarize the results of our experiments in Table I. CPU Times are given in CPU seconds on a Sun SPARCstation 1.



TABLE I. SUMMARY OF RESULTS OF ALGORITHM ON RANDOM INSTANCES

Type of Graph	Size	Num Trials	Ave Int Gap	Ave CPU Time
Type A	50	50	.96988	36.28
	100	20	.96783	323.08
	200	5	.97209	4629.62
Type B	50	50	.97202	23.06
	100	20	.97097	217.42
	200	5	.97237	2989.00
Type C	50	50	.95746	23.53
	100	20	.94214	306.84
	200	5	.92362	2546.42
Type D	50	50	.95855	27.35
	100	20	.93984	355.32
	200	5	.93635	10709.42

NOTE: Ave Int Gap is the average ratio of the value of the cut generated to the semidefinite bound, except for Type B graphs, where it is the ratio of the value of the cut generated minus the negative edge weights to the semidefinite bound minus the negative edge weights.

In the case of the TSP instances, we used Euclidean instances from the TSPLIB, and set the edge weight  $w_{ij}$  to the Euclidean distance between the points  $i$  and  $j$ . We summarize our results in Table II. In all 10 instances, we compute the optimal solution; for 5 instances, the value of the cut produced is equal to  $Z_p^*$ , and for the others, we have been able to exploit additional information from the dual semidefinite program to prove optimality (for the problems dantzig42, gr48 and hk48, Poljak and Rendl [1995b] also show that our solution is optimal). For all TSPLIB instances, the maximum cut value is within .995 of the semidefinite bound. Given these computational results, it is tempting to speculate that a much stronger bound can be proven for these Euclidean instances. However, the instance defined by a unit length equilateral triangle has a maximum cut value of 2, but  $Z_p^* = \frac{9}{4}$ , for a ratio of  $\frac{8}{9} = 0.8889$ .

Homer and Peinado [unpublished manuscript] have implemented our algorithm on a CM-5, and have shown that it produces optimal or very nearly optimal solutions to a number of MAX CUT instances derived from via minimization problems. These instances were provided by Michael Jünger (personal communication) and have between 828 and 1366 vertices.

## 7. Generalizations

We can use the same technique as in Section 2 to approximate several other problems. In the next section, we describe a variation of MAX CUT and give

TABLE II. SUMMARY OF RESULTS OF  
ALGORITHM ON TSPLIB INSTANCES

Instance	Size	SD Val	Cut Val	Time
dantzig42	42	42638	42638	43.35
gr120	120	2156775	2156667	754.87
gr48	48	321815	320277	26.17
gr96	96	105470	105295	531.50
hk48	48	771712	771712	66.52
kroA100	100	5897392	5897392	420.83
kroB100	100	5763047	5763047	917.47
kroC100	100	5890760	5890760	398.78
kroD100	100	5463946	5463250	469.48
kroE100	100	5986675	5986591	375.68

NOTE: SD Val is the value produced by the semidefinite relaxation. Cut Val is the value of the best cut output by the algorithm.

an  $(\alpha - \epsilon)$ -approximation algorithm for it. In Section 7.2, we give an  $(\alpha - \epsilon)$ -approximation algorithm for the MAX 2SAT problem, and show that it leads to a slightly improved algorithm for MAX SAT. Finally, in Section 7.3, we give a  $(\beta - \epsilon)$ -approximation algorithm for the maximum directed cut problem (MAX DICUT), where  $\beta > .79607$ . In all cases, we will show how to approximate more general integer quadratic programs that can be used to model these problems.

**7.1 MAX RES CUT.** The MAX RES CUT problem is a variation of MAX CUT in which pairs of vertices are forced to be on either the same side of the cut or on different sides of the cut. The extension of the algorithm to the MAX RES CUT problem is trivial. We merely need to add the following constraints to  $(P)$ :  $v_i \cdot v_j = 1$  for  $(i, j) \in E^+$  and  $v_i \cdot v_j = -1$  for  $(i, j) \in E^-$ , where  $E^+$  (respectively,  $E^-$ ) corresponds to the pair of vertices forced to be on the same side (respectively, different sides) of the cut. Using the randomized algorithm of Section 2 and setting  $y_i = 1$  if  $r \cdot v_i \geq 0$ , and  $y_i = -1$  otherwise, gives a feasible solution to MAX RES CUT, assuming that a feasible solution exists. Indeed, it is easy to see that if  $v_i \cdot v_j = 1$ , then the algorithm will produce a solution such that  $y_i y_j = 1$ . If  $v_i \cdot v_j = -1$ , then the only case in which the algorithm produces a solution such that  $y_i y_j \neq -1$  is when  $v_i \cdot r = v_j \cdot r = 0$ , an event that happens with probability 0. The analysis of the expected value of the cut is unchanged and, therefore, the resulting algorithm is a randomized  $(\alpha - \epsilon)$ -approximation algorithm.

Another approach to the problem is to use a standard reduction of MAX RES CUT to MAX CUT based on contracting edges and “switching” cuts (see,

e.g. [Poljak and Rendl to appear]). This reduction introduces negative edge weights and so we do not discuss it here, although Theorem 3.2.1 can be used to show that our MAX CUT algorithm applied to a reduced instance has a performance guarantee of  $(\alpha - \epsilon)$  for the original MAX RES CUT instance. In fact, a more general statement can be made: any  $\rho$ -approximation algorithm (in the sense of Theorem 3.2.1) for MAX CUT instances possibly having negative edge weights leads to a  $\rho$ -approximation algorithm for MAX RES CUT.

**7.2 MAX 2SAT AND MAX SAT.** An instance of the maximum satisfiability problem (MAX SAT) is defined by a collection  $\mathcal{C}$  of Boolean clauses, where each clause is a disjunction of literals drawn from a set of variables  $\{x_1, x_2, \dots, x_n\}$ . A *literal* is either a variable  $x$  or its negation  $\bar{x}$ . The *length*  $l(C_j)$  of a clause  $C_j$  is the number of distinct literals in the clause. In addition, for each clause  $C_j \in \mathcal{C}$  there is an associated nonnegative weight  $w_j$ . An optimal solution to a MAX SAT instance is an assignment of truth values to the variables  $x_1, \dots, x_n$  that maximizes the sum of the weight of the satisfied clauses. MAX 2SAT consists of MAX SAT instances in which each clause has length at most two. MAX 2SAT is NP-complete [Garey et al. 1976]; the best approximation algorithm known previously has a performance guarantee of  $\frac{3}{4}$  and is due to Yannakakis [1994] (see also Goemans and Williamson [1994b]). As with the MAX CUT problem, MAX 2SAT is known to be MAX SNP-hard [Papadimitriou and Yannakakis 1991]; thus, there exists some constant  $c < 1$  such that the existence of a  $c$ -approximation algorithm implies that  $P = NP$  [Arora et al. 1992]. Bellare et al. [unpublished manuscript] have shown that a 95/96-approximation algorithm for MAX 2SAT would imply  $P = NP$ . Haglin [1992] and Haglin (personal communication) has shown that any  $\rho$ -approximation algorithm for MAX RES CUT can be translated into a  $\rho$ -approximation algorithm for MAX 2SAT, but we will show a direct algorithm here. Haglin's observation together with the reduction from MAX RES CUT to MAX CUT mentioned in the previous section shows that any  $\rho$ -approximation for MAX CUT with negative edge weights translates into a  $\rho$ -approximation algorithm for MAX 2SAT.

**7.2.1 MAX 2SAT.** In order to model MAX 2SAT, we consider the integer quadratic program

$$\begin{aligned} & \text{Maximize} \quad \sum_{i < j} [a_{ij}(1 - y_i y_j) + b_{ij}(1 + y_i y_j)] \\ (Q') \quad & \text{subject to: } y_i \in \{-1, 1\} \quad \forall i \in V, \end{aligned}$$

where  $a_{ij}$  and  $b_{ij}$  are nonnegative. The objective function of  $(Q')$  is thus a nonnegative linear form in  $1 \pm y_i y_j$ . To model MAX 2SAT using  $(Q')$ , we introduce a variable  $y_i$  in the quadratic program for each Boolean variable  $x_i$  in the 2SAT instance; we also introduce an additional variable  $y_0$ . The value of  $y_0$  will determine whether  $-1$  or  $1$  will correspond to "true" in the MAX 2SAT instance. More precisely,  $x_i$  is true if  $y_i = y_0$  and false otherwise. Given a Boolean formula  $C$ , we define its value  $v(C)$  to be 1 if the formula is true

and 0 if the formula is false. Thus,

$$v(x_i) = \frac{1 + y_0 y_i}{2}$$

and

$$v(\bar{x}_i) = 1 - v(x_i) = \frac{1 - y_0 y_i}{2}.$$

Observe that

$$\begin{aligned} v(x_i \vee x_j) &= 1 - v(\bar{x}_i \wedge \bar{x}_j) = 1 - v(\bar{x}_i)v(\bar{x}_j) = 1 - \frac{1 - y_0 y_i}{2} \frac{1 - y_0 y_j}{2} \\ &= \frac{1}{4} (3 + y_0 y_i + y_0 y_j - y_0^2 y_i y_j) \\ &= \frac{1 + y_0 y_i}{4} + \frac{1 + y_0 y_j}{4} + \frac{1 - y_i y_j}{4}. \end{aligned}$$

The value of other clauses with 2 literals can be similarly expressed; for instance, if  $x_i$  is negated one only needs to replace  $y_i$  by  $-y_i$ . Therefore, the value  $v(C)$  of any clause with at most two literals per clause can be expressed in the form required in  $(Q')$ . As a result, the MAX 2SAT problem can be modelled as

$$\text{Maximize } \sum_{C_j \in \mathcal{C}} w_j v(C_j)$$

$$(SAT) \quad \text{subject to: } y_i \in \{-1, 1\} \quad \forall i \in \{0, 1, \dots, n\},$$

where the  $v(C_j)$  are nonnegative linear combinations of  $1 + y_i y_j$  and  $1 - y_i y_j$ . The  $(SAT)$  program is in the same form as  $(Q')$ .

We relax  $(Q')$  to:

$$\text{Maximize } \sum_{i < j} [a_{ij}(1 - v_i \cdot v_j) + b_{ij}(1 + v_i \cdot v_j)]$$

$$(P') \quad \text{subject to: } v_i \in S_n \quad \forall i \in V.$$

Let  $E[V]$  be the expected value of the solution produced by the randomized algorithm. By the linearity of expectation,

$$\begin{aligned} E[V] &= 2 \sum_{i < j} a_{ij} \Pr[\text{sgn}(v_i \cdot r) \neq \text{sgn}(v_j \cdot r)] \\ &\quad + 2 \sum_{i < j} b_{ij} \Pr[\text{sgn}(v_i \cdot r) = \text{sgn}(v_j \cdot r)]. \end{aligned}$$

Using the analysis of the max cut algorithm, we note that  $\Pr[\text{sgn}(v_i \cdot r) = \text{sgn}(v_j \cdot r)] = 1 - 1/\pi \arccos(v_i \cdot v_j)$ , and thus the approximation ratio for the more general program follows from Lemmas 3.4 and 3.2.2.

Hence, we can show the following theorem, which implies that the algorithm is an  $(\alpha - \epsilon)$ -approximation algorithm for  $(Q')$  and thus for MAX 2SAT.

## THEOREM 7.2.1.1

$$E[V] \geq \alpha \sum_{i < j} [a_{ij}(1 - v_i \cdot v_j) + b_{ij}(1 + v_i \cdot v_j)].$$

**7.2.2 MAX SAT.** The improved MAX 2SAT algorithm leads to a slightly improved approximation algorithm for MAX SAT. In Goemans and Williamson [1994b], we developed several randomized  $\frac{3}{4}$ -approximation algorithms for MAX SAT. We considered the following linear programming relaxation of the MAX SAT problem, where  $I_j^+$  denotes the set of nonnegated variables in clause  $C_j$  and  $I_j^-$  is the set of negated variables in  $C_j$ :

$$\begin{aligned} &\text{Max} && \sum_{C_j \in \mathcal{C}} w_j z_j \\ &\text{subject to:} && \sum_{i \in I_j^+} y_i + \sum_{i \in I_j^-} (1 - y_i) \geq z_j \quad \forall C_j \in \mathcal{C} \\ &&& 0 \leq y_i \leq 1 \quad 1 \leq i \leq n \\ &&& 0 \leq z_j \leq 1 \quad \forall C_j \in \mathcal{C}. \end{aligned}$$

By associating  $y_i = 1$  with  $x_i$  set true,  $y_i = 0$  with  $x_i$  set false,  $z_j = 1$  with clause  $C_j$  satisfied, and  $z_j = 0$  with clause  $C_j$  not satisfied, the program exactly corresponds to the MAX SAT problem. We showed that for any feasible solution  $(y, z)$ , if  $x_i$  is set to be true with probability  $y_i$ , then the probability that clause  $j$  will be satisfied is at least

$$\left(1 - \left(1 - \frac{1}{k}\right)^k\right) z_j,$$

for  $k = l(C_j)$ . We then considered choosing randomly between the following two algorithms: (1) set  $x_i$  true independently with probability  $y_i$ ; (2) set  $x_i$  true independently with probability  $\frac{1}{2}$ . Given this combined algorithm, the probability that a length  $k$  clause is satisfied is at least

$$\frac{1}{2}(1 - 2^{-k}) + \frac{1}{2}\left(1 - \left(1 - \frac{1}{k}\right)^k\right) z_j.$$

This expression can be shown to be at least  $\frac{3}{4}z_j$  for all lengths  $k$ . Thus, if an optimal solution to the linear program is used, the algorithm results in a  $\frac{3}{4}$ -approximation algorithm for MAX SAT, since the expected value of the algorithm is at least  $\frac{3}{4} \sum_j w_j z_j$ .

We formulate a slightly different relaxation of the MAX SAT problem. Let  $u(C)$  denote a relaxed version of the expression  $v$  used in the previous section in which the products  $y_i y_j$  are replaced by inner products of vectors  $v_i \cdot v_j$ . Thus,

$$u(x_i) = \frac{1}{2}(1 + v_i \cdot v_0), \quad u(\bar{x}_i) = \frac{1}{2}(1 - v_i \cdot v_0),$$

and

$$u(x_i \vee x_j) = \frac{1}{4}(1 + v_i \cdot v_0) + \frac{1}{4}(1 + v_j \cdot v_0) + \frac{1}{4}(1 - v_i \cdot v_j).$$

We then consider the following relaxation,

$$\begin{aligned} & \text{Max} && \sum_{C_j \in \mathcal{C}} w_j z_j \\ & \text{subject to:} && \\ & && \sum_{i \in I_j^+} u(x_i) + \sum_{i \in I_j^-} u(\bar{x}_i) \geq z_j \quad \forall C_j \in \mathcal{C} \\ & && u(C_j) \geq z_j \quad \forall C_j \in \mathcal{C}, l(C_j) = 2 \\ & && v_i \cdot v_i = 1 \quad 0 \leq i \leq n \\ & && 0 \leq z_j \leq 1 \quad \forall C_j \in \mathcal{C}. \end{aligned}$$

Thus, if we set  $x_i$  to be true with probability  $u(x_i)$  for the optimal solution to the corresponding semidefinite program, then by the arguments of Goemans and Williamson [1994b], we satisfy a clause  $C_j$  of length  $k$  with probability at least  $(1 - (1 - (1/k))^k)z_j$ .

To obtain the improved bound, we consider three algorithms: (1) set  $x_i$  true independently with probability  $\frac{1}{2}$ ; (2) set  $x_i$  true independently with probability  $u(x_i)$  (given the optimal solution to the program); (3) pick a random unit vector  $r$  and set  $x_i$  true iff  $\text{sgn}(v_i \cdot r) = \text{sgn}(v_0 \cdot r)$ . Suppose we use algorithm  $i$  with probability  $p_i$ , where  $p_1 + p_2 + p_3 = 1$ . From the previous section, for algorithm (3) the probability that a clause  $C_j$  of length 1 or 2 is satisfied is at least  $\alpha u(C_j) \geq \alpha z_j$ . Thus the expected value of the solution is at least

$$\begin{aligned} & \sum_{j: l(C_j)=1} w_j (.5p_1 + (p_2 + \alpha p_3)z_j) + \sum_{j: l(C_j)=2} w_j (.75p_1 + (.75p_2 + \alpha p_3)z_j) \\ & + \sum_{j: l(C_j) \geq 3} w_j \left( (1 - 2^{-l(C_j)})p_1 + \left( 1 - \left( 1 - \frac{1}{l(C_j)} \right)^{l(C_j)} \right) p_2 z_j \right). \end{aligned}$$

If we set  $p_1 = p_2 = .4785$  and  $p_3 = .0430$ , then the expected value is at least  $.7554 \sum_j w_j z_j$ , yielding a .7554-approximation algorithm. To see this, we check the value of the expression for lengths 1 through 4, and notice that

$$\min_k \left( 1 - \left( 1 - \frac{1}{k} \right)^k \right) \geq 1 - \frac{1}{e}$$

and

$$.4785 \left( (1 - 2^{-5}) + 1 - \frac{1}{e} \right) \geq .7554.$$

We can obtain even a slightly better approximation algorithm for the MAX SAT problem. The bottleneck in the analysis above is that algorithm (3) contributes no expected weight for clauses of length 3 or greater. For a given clause  $C_j$  of length 3 or more, let  $P_j$  be a set of length 2 clauses formed by

taking the literals of  $C_j$  two at a time; thus,  $P_j$  will contain  $\binom{l(C_j)}{2}$  clauses. If at least one of the literals in  $C_j$  is set true, then at least  $l(C_j) - 1$  of the clauses in  $P_j$  will be satisfied. Thus, the following program is a relaxation of the MAX SAT problem:

$$\begin{aligned}
 &\text{Max} && \sum_{C_j \in \mathcal{C}} w_j z_j \\
 &\text{subject to:} && \\
 &&& \sum_{i \in I_j^+} u(x_i) + \sum_{i \in I_j^-} u(\bar{x}_i) \geq z_j && \forall C_j \in \mathcal{C} \\
 &&& u(C_j) \geq z_j && \forall C_j \in \mathcal{C}, l(C_j) = 2 \\
 &&& \frac{1}{l(C_j) - 1} \sum_{C \in P_j} u(C) \geq z_j && \forall C_j \in \mathcal{C}, l(C_j) \geq 3 \\
 &&& v_i \cdot v_i = 1 && 0 \leq i \leq n \\
 &&& 0 \leq z_j \leq 1 && \forall C_j \in \mathcal{C}.
 \end{aligned}$$

Algorithm (3) has expected value of  $\alpha u(C)$  for each  $C \in P_j$  for any  $j$ , so that its expected value for any clause of length 3 or more becomes at least

$$\begin{aligned}
 \alpha \cdot \frac{1}{\binom{l(C_j)}{2}} \sum_{C \in P_j} u(C) &= \alpha \cdot \frac{2}{l(C_j)} \frac{1}{l(C_j) - 1} \sum_{C \in P_j} u(C) \\
 &\geq \alpha \cdot \frac{2}{l(C_j)} z_j,
 \end{aligned}$$

so that the overall expectation of the algorithm will be at least

$$\begin{aligned}
 &\sum_{j: l(C_j)=1} w_j (.5p_1 + (p_2 + \alpha p_3)z_j) + \sum_{j: l(C_j)=2} w_j (.75p_1 + (.75p_2 + \alpha p_3)z_j) \\
 &+ \sum_{j: l(C_j) \geq 3} w_j \left( (1 - 2^{-l(C_j)})p_1 \right. \\
 &\quad \left. + \left( \left( 1 - \left( 1 - \frac{1}{l(C_j)} \right)^{l(C_j)} \right) p_2 + \alpha \cdot \frac{2}{l(C_j)} p_3 \right) z_j \right).
 \end{aligned}$$

By setting  $p_1 = p_2 = .467$  and  $p_3 = .066$ , we obtain a .7584-approximation algorithm, which can be verified by checking the expression for lengths 1 through 6, and noticing that

$$.467 \left( \left( 1 - \frac{1}{e} \right) + (1 - 2^{-7}) \right) \geq .7584.$$

Other small improvements are possible by tightening the analysis.

**7.3 MAX DICUT.** Suppose we are given a directed graph  $G = (V, A)$  and weights  $w_{ij}$  on each directed arc  $(i, j) \in A$ , where  $i$  is the *tail* of the arc and  $j$  is the *head*. The maximum directed cut problem is that of finding the set of vertices  $S$  that maximizes the weight of the edges with their tails in  $S$  and their heads in  $\bar{S}$ . The problem is NP-hard via a straightforward reduction from MAX CUT. The best previously known approximation algorithm for MAX DICUT has a performance guarantee of  $\frac{1}{4}$  [Papadimitriou and Yannakakis 1991].

To model MAX DICUT, we consider the integer quadratic program

$$\begin{aligned} \text{Max} \quad & \sum_{i,j,k} [c_{ijk}(1 - y_i y_j - y_i y_k + y_j y_k) \\ & + d_{ijk}(1 - y_i y_j + y_i y_k + y_j y_k)] \\ (Q'') \quad & \text{subject to: } y_i \in \{-1, 1\} \quad \forall i \in V, \end{aligned}$$

where  $c_{ijk}$  and  $d_{ijk}$  are nonnegative. Observe that  $1 - y_i y_j - y_i y_k + y_j y_k$  can also be written as  $(1 - y_i y_j)(1 - y_i y_k)$  (or as  $(1 - y_i y_j)(1 + y_j y_k)$ ), and, thus, the objective function of  $(Q'')$  can be interpreted as a nonnegative restricted quadratic form in  $1 \pm y_i y_j$ . Moreover,  $1 - y_i y_j - y_i y_k + y_j y_k$  is equal to 4 if  $y_i = -y_j = -y_k$  and 0 otherwise, while  $1 + y_i y_j + y_i y_k + y_j y_k$  is 4 if  $y_i = y_j = y_k$  and is 0 otherwise.

We can model the MAX DICUT problem using the program  $(Q'')$  by introducing a variable  $y_i$  for each  $i \in V$ , and, as with the MAX 2SAT program, and introducing a variable  $y_0$  that will denote the  $S$  side of the cut. Thus,  $i \in S$  iff  $y_i = y_0$ . Then arc  $(i, j)$  contributes weight

$$\frac{1}{4} w_{ij} (1 + y_i y_0)(1 - y_j y_0) = \frac{1}{4} w_{ij} (1 + y_i y_0 - y_j y_0 - y_i y_j)$$

to the cut. Summing over all arcs  $(i, j) \in A$  gives a program of the same form as  $(Q'')$ . We observe that if the directed graph has weighted indegree of every vertex equal to weighted outdegree, the program  $(Q'')$  reduces to one of the form  $(Q')$ , and therefore our approximation algorithm has a performance guarantee of  $(\alpha - \epsilon)$ .

We relax  $(Q'')$  to:

$$\begin{aligned} \text{Max} \quad & \sum_{i,j,k} [c_{ijk}(1 - v_i \cdot v_j - v_i \cdot v_k + v_j \cdot v_k) \\ & + d_{ijk}(1 + v_i \cdot v_j + v_i \cdot v_k + v_j \cdot v_k)] \\ (P'') \quad & \text{subject to: } v_i \in S_n \quad \forall i \in V. \end{aligned}$$

We approximate  $(Q'')$  by using exactly the same algorithm as before. The analysis is somewhat more complicated. As we will show, the performance guarantee  $\beta$  is slightly weaker, namely

$$\beta = \min_{0 \leq \theta < \arccos(-1/3)} \frac{2}{\pi} \frac{2\pi - 3\theta}{1 + 3\cos\theta} > 0.79607.$$



Given a vector  $r$  drawn uniformly from the unit sphere  $S_n$ , we know by the linearity of expectation that the expected value  $E[U]$  of the solution output is

$$4 \sum_{i,j,k} \left[ c_{ijk} \cdot \Pr[\text{sgn}(v_i \cdot r) \neq \text{sgn}(v_j \cdot r) = \text{sgn}(v_k \cdot r)] \right. \\ \left. + d_{ijk} \cdot \Pr[\text{sgn}(v_i \cdot r) = \text{sgn}(v_j \cdot r) = \text{sgn}(v_k \cdot r)] \right].$$

Consider any term in the sum, say  $d_{ijk} \cdot \Pr[\text{sgn}(v_i \cdot r) = \text{sgn}(v_j \cdot r) = \text{sgn}(v_k \cdot r)]$ . The  $c_{ijk}$  terms can be dealt with similarly by simply replacing  $v_i$  by  $-v_i$ . The performance guarantee follows from the proof of the following two lemmas.

LEMMA 7.3.1

$$\Pr[\text{sgn}(v_i \cdot r) = \text{sgn}(v_j \cdot r) = \text{sgn}(v_k \cdot r)] \\ = 1 - \frac{1}{2\pi} (\arccos(v_i \cdot v_j) + \arccos(v_i \cdot v_k) + \arccos(v_j \cdot v_k)).$$

LEMMA 7.3.2. For any  $v_i, v_j, v_k \in S_n$ ,

$$1 - \frac{1}{2\pi} (\arccos(v_i \cdot v_j) + \arccos(v_i \cdot v_k) + \arccos(v_j \cdot v_k)) \\ \geq \frac{\beta}{4} [1 + v_i \cdot v_j + v_i \cdot v_k + v_j \cdot v_k],$$

where  $\beta = \min_{0 \leq \theta < \arccos(-1/3)} \frac{2}{\pi} \frac{2\pi - 3\theta}{1 + 3 \cos \theta} > 0.79607$ .

PROOF OF LEMMA 7.3.1. A very short proof can be given relying on spherical geometry. The desired probability can be seen to be equal to twice the area of the spherical triangle polar to the spherical triangle defined by  $v_i, v_j$ , and  $v_k$ . Stated this way, the result is a corollary to Girard's [1629] formula (see Rosenfeld [1988]) expressing the area of a spherical triangle with angles  $\theta_1, \theta_2$ , and  $\theta_3$  as its excess  $\theta_1 + \theta_2 + \theta_3 - \pi$ .

We also present a proof of the lemma from first principles. In fact, our proof parallels Euler's [1781] proof (see Rosenfeld [1988]) of Girard's formula. We define the following events:

$$\begin{aligned} A: & \text{sgn}(v_i \cdot r) = \text{sgn}(v_j \cdot r) = \text{sgn}(v_k \cdot r) \\ B_i: & \text{sgn}(v_i \cdot r) \neq \text{sgn}(v_j \cdot r) = \text{sgn}(v_k \cdot r) \\ C_i: & \text{sgn}(v_j \cdot r) = \text{sgn}(v_k \cdot r) \\ C_j: & \text{sgn}(v_i \cdot r) = \text{sgn}(v_k \cdot r) \\ C_k: & \text{sgn}(v_i \cdot r) = \text{sgn}(v_j \cdot r). \end{aligned}$$

Note that  $B_i = C_i - A$ . We define  $B_j$  and  $B_k$  similarly, so that  $B_j = C_j - A$  and  $B_k = C_k - A$ . Clearly,

$$\Pr[A] + \Pr[B_i] + \Pr[B_j] + \Pr[B_k] = 1. \quad (1)$$

Also,  $\Pr[C_i] = \Pr[A] + \Pr[B_i]$  and similarly for  $j$  and  $k$ . Adding up these equalities and subtracting (1), we obtain

$$\Pr[C_i] + \Pr[C_j] + \Pr[C_k] = 1 + 2 \Pr[A]. \quad (2)$$

By Lemma 3.2,  $\Pr[C_i] = 1 - 1/\pi \arccos(v_i \cdot v_k)$  and similarly for  $j$  and  $k$ . Together with (2), we derive

$$\Pr[A] = 1 - \frac{1}{2\pi} (\arccos(v_i \cdot v_j) + \arccos(v_i \cdot v_k) + \arccos(v_j \cdot v_k)),$$

proving the lemma.  $\square$

PROOF OF LEMMA 7.3.2. One can easily verify that the defined value of  $\beta$  is greater than 0.79607. Let  $a = \arccos(v_i \cdot v_j)$ ,  $b = \arccos(v_i \cdot v_k)$ , and  $c = \arccos(v_j \cdot v_k)$ . From the theory of spherical triangles, it follows that the possible values for  $(a, b, c)$  over all possible vectors  $v_i, v_j$ , and  $v_k$  define the set

$$S = \{(a, b, c) : 0 \leq a \leq \pi, 0 \leq b \leq \pi, 0 \leq c \leq \pi, \\ c \leq a + b, b \leq a + c, a \leq b + c, a + b + c \leq 2\pi\}.$$

(see Berger [1987, Corollary 18.6.12.3]). The claim can thus be restated as

$$1 - \frac{1}{2\pi}(a + b + c) \geq \frac{\beta}{4}(1 + \cos(a) + \cos(b) + \cos(c))$$

for all  $(a, b, c) \in S$ .

Let  $(a, b, c)$  minimize

$$h(a, b, c) = 1 - \frac{1}{2\pi}(a + b + c) - \frac{\beta}{4}(1 + \cos(a) + \cos(b) + \cos(c))$$

over all  $(a, b, c) \in S$ . We consider several cases:

- (1)  $a + b + c = 2\pi$ . We have  $1 - (1/2\pi)(a + b + c) = 0$ . On the other hand,

$$\begin{aligned} 1 + \cos(a) + \cos(b) + \cos(c) &= 1 + \cos(a) + \cos(b) + \cos(a + b) \\ &= 1 + \cos(a + b) + 2 \cos\left(\frac{a + b}{2}\right) \cos\left(\frac{a - b}{2}\right) \\ &= 2 \cos^2\left(\frac{a + b}{2}\right) + 2 \cos\left(\frac{a + b}{2}\right) \cos\left(\frac{a - b}{2}\right) \\ &= 2 \cos\left(\frac{a + b}{2}\right) \left[ \cos\left(\frac{a + b}{2}\right) + \cos\left(\frac{a - b}{2}\right) \right]. \end{aligned} \quad (3)$$

We now derive that

$$h(a, b, c) \geq -\frac{\beta}{2} \cos\left(\frac{a + b}{2}\right) \left[ \cos\left(\frac{a + b}{2}\right) + \cos\left(\frac{a - b}{2}\right) \right] \geq 0,$$

the last inequality following from the fact that

$$\frac{\pi}{2} \leq \frac{a + b}{2} \leq \pi - \left| \frac{a - b}{2} \right|$$

and thus

$$\cos\left(\frac{a+b}{2}\right) \leq 0 \quad \text{and} \quad \left[\cos\left(\frac{a+b}{2}\right) + \cos\left(\frac{a-b}{2}\right)\right] \geq 0.$$

- (2)  $a = b + c$  or  $b = a + c$  or  $c = a + b$ . By symmetry, assume that  $c = a + b$ . Observe that

$$1 - \frac{1}{2\pi}(a + b + c) = 1 - \frac{a + b}{\pi}.$$

On the other hand, by (3) we have that

$$\begin{aligned} & 1 + \cos(a) + \cos(b) + \cos(c) \\ &= 2 \cos\left(\frac{a+b}{2}\right) \left(\cos\left(\frac{a-b}{2}\right) + \cos\left(\frac{a+b}{2}\right)\right) \\ &\leq 2 \cos\left(\frac{a+b}{2}\right) \left(1 + \cos\left(\frac{a+b}{2}\right)\right). \end{aligned}$$

Letting  $x = (a + b)/2$ , we observe that the claim is equivalent to

$$1 - \frac{2x}{\pi} \geq \frac{\beta}{2} \cos(x)(1 + \cos(x))$$

for any  $0 \leq x \leq \pi/2$ . One can in fact verify that

$$1 - \frac{2x}{\pi} \geq \frac{0.81989}{2} \cos(x)(1 + \cos(x))$$

for any  $0 \leq x \leq \pi/2$ , implying the claim.

- (3)  $a = 0$  or  $b = 0$  or  $c = 0$ . Without loss of generality, let  $a = 0$ . The definition of  $S$  implies that  $b = c$ , and thus  $b = a + c$ . This case therefore reduces to the previous one.
- (4)  $a = \pi$  or  $b = \pi$  or  $c = \pi$ . Assume  $a = \pi$ . This implies that  $b + c = \pi$  and, thus,  $a + b + c = 2\pi$ . We have thus reduced the problem to case (1).
- (5) In the last case,  $(a, b, c)$  belongs to the interior of  $S$ . This implies that the gradient of  $h$  must vanish and the hessian of  $h$  must be positive semidefinite at  $(a, b, c)$ . In other words,

$$\sin a = \sin b = \sin c = \frac{2}{\beta\pi},$$

and  $\cos a \geq 0$ ,  $\cos b \geq 0$  and  $\cos c \geq 0$ . From this, we derive that  $a = b = c$ . But

$$h(a, a, a) = 1 - \frac{3a}{2\pi} - \frac{\beta}{4}(1 + 3\cos(a)).$$

The lemma now follows from the fact that  $a \leq 2\pi/3$ , the definition of  $\beta$  and the fact that  $1 + 3\cos \alpha \leq 0$  for  $\alpha \geq \arccos -1/3$ .  $\square$

Thus, we obtain a  $(\beta - \epsilon)$ -approximation algorithm for  $(Q'')$  and for the MAX DICUT problem.

### 8. *Concluding Remarks*

Our motivation for studying semidefinite programming relaxations came from a realization that the standard tool of using linear programming relaxations for approximation algorithms had limits which might not be easily surpassed (see the conclusion of Goemans and Williamson [1994b]). In fact, a classical linear programming relaxation for the maximum cut problem can be shown to be arbitrarily close to twice the value of the maximum cut in the worst case. Given the work of Lovász and Schrijver [1989; 1990], which showed that tighter and tighter relaxations could be obtained through semidefinite programming, it seemed worthwhile to investigate the power of such relaxations from a worst-case perspective. The results of this paper constitute a first step in this direction. As we mentioned in the introduction, further steps have already been made, with improved results for MAX 2SAT and MAX DICUT by Feige and Goemans, and for coloring by Karger, Motwani, and Sudan. We think that the continued investigation of these methods is promising.

While this paper leaves many open questions, we think there are two especially interesting problems. The first question is whether a .878-approximation algorithm for MAX CUT can be obtained without explicitly solving the semidefinite program. For example, the first 2-approximation algorithms for weighted vertex cover involved solving a linear program [Hochbaum 1982], but later Bar-Yehuda and Even [1981] devised a primal-dual algorithm in which linear programming was used only in the analysis of the algorithm. Perhaps a semidefinite analog is possible for MAX CUT. The second question is whether adding additional constraints to the semidefinite program leads to a better worst-case bound. There is some reason to think this might be true. Linear constraints are known for which the program would find an optimal solution on any planar graph, whereas there is a gap of  $32/(25 + 5\sqrt{5})$  for the current semidefinite program for the 5-cycle.

One consequence of this paper is that the situation with several MAX SNP problems is no longer clear-cut. When the best-known approximation results for MAX CUT and MAX SAT had such long-standing and well-defined bounds as  $\frac{1}{2}$  and  $\frac{3}{4}$ , it was tempting to believe that perhaps no further work could be done in approximating these problems, and that it was only a matter of time before matching hardness results would be found. The improved results in this paper should rescue algorithm designers from such fatalism. Although MAX SNP problems cannot be approximated arbitrarily closely, there still is work to do in designing improved approximation algorithms.

**ACKNOWLEDGMENTS.** Since the appearance of an extended abstract of this paper, we have benefited from discussions with a very large number of colleagues. In particular, we would like to thank Don Coppersmith and David Shmoys for pointing out to us that our MAX 2SAT algorithm could lead to an improved MAX SAT algorithm, Jon Kleinberg for bringing Lovász [1983] to our attention, Bob Vanderbei for kindly giving us his semidefinite code, Francisco Barahona and Michael Jünger for providing problem instances, Joel Spencer for motivating Theorem 3.1.1, Farid Alizadeh, Gabor Pataki, and Rob Freund for results on semidefinite programming, and Shang-Hua Teng for bringing Knuth [1981] to our attention. We received other useful comments from Farid Alizadeh, Joseph Cheriyan, Jon Kleinberg, Monique Laurent, Colin

McDiarmid, Giovanni Rinaldi, David Shmoys, Éva Tardos, and the two anonymous referees.

## REFERENCES

- ALIZADEH, F. 1995. Interior point methods in semidefinite programming with applications to combinatorial optimization. *SIAM J. Optimiz.* 5, 13–51.
- ARORA, S., LUND, C., MOTWANI, R., SUDAN, M., AND SZEGEDY, M. 1992. Proof verification and hardness of approximation problems. In *Proceedings of the 33rd Annual Symposium on Foundations of Computer Science*. IEEE, New York, pp. 14–23.
- BAR-YEHUDA, R., AND EVEN, S. 1981. A linear time approximation algorithm for the weighted vertex cover problem. *J. Algorithms* 2, 198–203.
- BARAHONA, F., GRÖTSCHEL, M., JÜNGER, M., AND REINELT, G. 1988. An application of combinatorial optimization to statistical physics and circuit layout design. *Oper. Res.* 36, 493–513.
- BARVINOK, A. I. 1995. Problems of distance geometry and convex properties of quadratic maps. *Disc. Computat. Geom.* 13, 189–202.
- BELLARE, M., GOLDREICH, O., AND SUDAN, M. 1995. Free bits, PCP and non-approximability—Towards tight results. In *Proceedings of the 36th Annual Symposium on Foundations of Computer Science*. IEEE, Los Alamitos, Calif., pp. 422–431.
- BERGER, M. 1987. *Geometry II*. Springer-Verlag, Berlin.
- BONDY, J. A., AND MURTY, U. S. R. 1976. *Graph Theory with Applications*. American Elsevier Publishing, New York, N.Y.
- BOYD, S., EL GHAOU, L., FERON, E., AND BALAKRISHNAN, V. 1994. *Linear Matrix Inequalities in System and Control Theory*. SIAM, Philadelphia, Pa.
- CHOR, B., AND SUDAN, M. 1995. A geometric approach to betweenness. In *Algorithms-ESA '95*, P. Spirakis, ed. Lecture Notes in Computer Science, vol. 979. Springer-Verlag, New York, pp. 227–237.
- CHRISTENSEN, J., AND VESTERSTRØM, J. 1979. A note on extreme positive definite matrices. *Math. Annalen*, 244:65–68.
- DELORME, C., AND POLJAK, S. 1993a. Combinatorial properties and the complexity of a max-cut approximation. *Europ. J. Combinat.* 14, 313–333.
- DELORME, C., AND POLJAK, S. 1993b. Laplacian eigenvalues and the maximum cut problem. *Math. Prog.* 62, 557–574.
- DELORME, C., AND POLJAK, S. 1993c. The performance of an eigenvalue bound on the max-cut problem in some classes of graphs. *Disc. Math.* 111, 145–156.
- EULER, L. 1781. *De mensura angulorum solidorum*. Petersburg.
- FEIGE, U., AND GOEMANS, M. X. 1995. Approximating the value of two prover proof systems, with applications to MAX 2SAT and MAX DICUT. In *Proceedings of the 3rd Israel Symposium on Theory of Computing and Systems*. IEEE Computer Society Press, Los Alamitos, Calif., pp. 182–189.
- FEIGE, U., AND LOVÁSZ, L. 1992. Two-prover one-round proof systems: Their power and their problems. In *Proceedings of the 24th Annual ACM Symposium on the Theory of Computing* (Victoria, S.C., Canada, May 4–6). ACM, New York, pp. 733–744.
- FRIEZE, A., AND JERRUM, M. 1996. Improved approximation algorithms for MAX  $k$ -CUT and MAX BISECTION. *Algorithmica*, to appear.
- GAREY, M., JOHNSON, D., AND STOCKMEYER, L. 1976. Some simplified NP-complete graph problems. *Theoret. Comput. Sci.* 1, 237–267.
- GIRARD, A. 1629. De la mesure de la superficie des triangles et polygones sphériques. Appendix to “Invention nouvelle en l’algèbre”. Amsterdam.
- GOEMANS, M. X., AND WILLIAMSON, D. P. 1994a. .878-approximation algorithms for MAX CUT and MAX 2SAT. In *Proceedings of the 26th Annual ACM Symposium on the Theory of Computing* (Montreal, Que., Canada, May 23–25). ACM, New York, pp. 422–431.
- GOEMANS, M. X., AND WILLIAMSON, D. P. 1994b. New  $3/4$ -approximation algorithms for the maximum satisfiability problem. *SIAM J. Disc. Math.* 7, 656–666.
- GOLUB, G. H., AND VAN LOAN, C. F. 1983. *Matrix Computations*. The Johns Hopkins University Press, Baltimore, Md.
- GRONE, R., PIERCE, S., AND WATKINS, W. 1990. Extremal correlation matrices. *Lin. Algebra Appl.* 134, 63–70.
- GRÖTSCHEL, M., LOVÁSZ, L., AND SCHRIJVER, A. 1981. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica* 1, 169–197.

- GRÖTSCHEL, M., LOVÁSZ, L., AND SCHRIJVER, A. 1988. *Geometric Algorithms and Combinatorial Optimization*. Springer-Verlag, Berlin.
- HADLOCK, F. 1975. Finding a maximum cut of a planar graph in polynomial time. *SIAM J. Comput.* 4, 221–225.
- HAGLIN, D. J. 1992. Approximating maximum 2-CNF satisfiability. *Paral. Proc. Lett.* 2, 181–187.
- HAGLIN, D. J., AND VENKATESAN, S. M. 1991. Approximation and intractability results for the maximum cut problem and its variants. *IEEE Trans. Comput.* 40, 110–113.
- HOCHBAUM, D. S. 1982. Approximation algorithms for the set covering and vertex cover problems. *SIAM J. Comput.* 11, 555–556.
- HOFMEISTER, T., AND LEFMANN, H. 1995. A combinatorial design approach to MAXCUT. In *Proceedings of the 13th Symposium on Theoretical Aspects of Computer Science*. To appear.
- HOMER, S., AND PEINADO, M. A highly parallel implementation of the Goemans/Williamson algorithm to approximate MaxCut. Manuscript.
- KARGER, D., MOTWANI, R., AND SUDAN, M. 1994. Approximate graph coloring by semidefinite programming. In *Proceedings of the 35th Annual Symposium on Foundations of Computer Science*. IEEE, New York, pp. 2–13.
- KARP, R. M. 1972. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, R. Miller and J. Thatcher, eds. Plenum Press, New York, pp. 85–103.
- KNUTH, D. E. 1981. *Seminumerical Algorithms*. Vol. 2 of *The Art of Computer Programming*. Addison-Wesley, Reading, Mass.
- LANCASTER, P., AND TISMENETSKY, M. 1985. *The Theory of Matrices*. Academic Press, Orlando, Fla.
- LAURENT, M., AND POLJAK, S. 1996. On the facial structure of the set of correlation matrices. *SIAM J. Matrix Anal. and Appl.* to appear.
- LI, C.-K., AND TAM, B.-S. 1994. A note on extreme correlation matrices. *SIAM J. Matrix Anal. and Appl.* 15, 903–908.
- LOEWY, R. 1980. Extreme points of a convex subset of the cone of positive definite matrices. *Math. Annalen.* 253, 227–232.
- LOVÁSZ, L. 1979. On the Shannon capacity of a graph. *IEEE Trans. Inf. Theory* IT-25, 1–7.
- LOVÁSZ, L. 1983. Self-dual polytopes and the chromatic number of distance graphs on the sphere. *Acta Sci. Math.* 45, 317–323.
- LOVÁSZ, L. 1992. Combinatorial optimization: Some problems and trends. DIMACS Technical Report 92-53.
- LOVÁSZ, L., AND SCHRIJVER, A. 1989. Matrix cones, projection representations, and stable set polyhedra. In *Polyhedral Combinatorics*, vol. 1 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*. American Mathematical Society, Providence, R.I.
- LOVÁSZ, L., AND SCHRIJVER, A. 1990. Cones of matrices and setfunctions, and 0–1 optimization. *SIAM J. Optimiz.* 1, 166–190.
- MAHAJAN, S., AND RAMESH, H. 1995. Derandomizing semidefinite programming based approximation algorithms. In *Proceedings of the 36th Annual Symposium on Foundations of Computer Science*. IEEE, Los Alamitos, Calif., pp. 162–163.
- MOHAR, B., AND POLJAK, S. 1993. Eigenvalue methods in combinatorial optimization. In *Combinatorial and Graph-Theoretic Problems in Linear Algebra*, vol. 50 of *The IMA Volumes in Mathematics and Its Applications*. R. Brualdi, S. Friedland, and V. Klee, eds. Springer-Verlag, New York.
- NESTEROV, Y., AND NEMIROVSKII, A. 1989. *Self-Concordant Functions and Polynomial Time Methods in Convex Programming*. Central Economic and Mathematical Institute, Academy of Science, Moscow, CIS.
- NESTEROV, Y., AND NEMIROVSKII, A. 1994. *Interior Point Polynomial Methods in Convex Programming*. Society for Industrial and Applied Mathematics, Philadelphia, Pa.
- ORLOVA, G. I., AND DORFMAN, Y. G. 1972. Finding the maximal cut in a graph. *Eng. Cyber.* 10, 502–506.
- OVERTON, M. L., AND WOMERSLEY, R. S. 1992. On the sum of the largest eigenvalues of a symmetric matrix. *SIAM J. Matrix Anal. and Appl.* 13, 41–45.
- OVERTON, M. L., AND WOMERSLEY, R. S. 1993. Optimality conditions and duality theory for minimizing sums of the largest eigenvalues of symmetric matrices. *Math. Prog.* 62, 321–357.
- PAPADIMITRIOU, C. H., AND YANNAKAKIS, M. 1991. Optimization, approximation, and complexity classes. *J. Comput. Syst. Sci.* 43, 425–440.
- PATAKI, G. 1994. On the multiplicity of optimal eigenvalues. Management Science Research Report MSRR-604, GSIA. Carnegie-Mellon Univ., Pittsburgh, Pa.

- PATAKI, G. 1995. On cone-LP's and semi-definite programs: Facial structure, basic solutions, and the simplex method. GSIA Working paper WP 1995-03, Carnegie-Mellon Univ., Pittsburgh, Pa.
- POLJAK, S., AND RENDL, F. 1994. Node and edge relaxations of the max-cut problem. *Computing* 52, 123–137.
- POLJAK, S., AND RENDL, F. 1995a. Nonpolyhedral relaxations of graph-bisection problems. *SIAM J. Optim.*, to appear.
- POLJAK, S., AND RENDL, F. 1995b. Solving the max-cut problem using eigenvalues. *Disc. Appl. Math.* 62, 249–278.
- POLJAK, S., AND TURŽÍK, D. 1982. A polynomial algorithm for constructing a large bipartite subgraph, with an application to a satisfiability problem. *Can. J. Math.* 34, 519–524.
- POLJAK, S., AND TUZA, Z. 1995. Maximum cuts and largest bipartite subgraphs. In *Combinatorial Optimization*. W. Cook, L. Lovász, and P. Seymour, eds. DIMACS series in Discrete Mathematics and Theoretical Computer Science, Vol. 20. American Mathematical Society, Providence, R.I. To be published.
- REINELT, G. 1991. TSPLIB—A traveling salesman problem library. *ORSA J. Comput.* 3, 376–384.
- RENDL, F., VANDERBEL, R., AND WOLKOWICZ, H. 1993. Interior point methods for max-min eigenvalue problems. Report 264, Technische Universität Graz, Graz, Austria.
- ROSENFELD, B. 1988. *A history of non-Euclidean geometry*. Springer-Verlag, Berlin.
- SAHNI, S., AND GONZALEZ, T. 1976. P-complete approximation problems. *J. ACM* 23, 3 (July), 555–565.
- VAIDYA, P. M. 1989. A new algorithm for minimizing convex functions over convex sets. In *Proceedings of the 30th Annual Symposium on Foundations of Computer Science*. IEEE, New York, pp. 338–343.
- VANDENBERGHE, L., AND BOYD, S. 1996. Semidefinite programming. *SIAM Rev.* To appear.
- VITÁNYI, P. M. 1981. How well can a graph be  $n$ -colored? *Disc. Math.* 34, 69–80.
- WOLKOWICZ, H. 1981. Some applications of optimization in matrix theory. *Lin. Algebra Appl.* 40, 101–118.
- YANNAKAKIS, M. 1994. On the approximation of maximum satisfiability. *J. Algorithms* 17, 475–502.

RECEIVED JUNE 1994; REVISED JUNE 1995; ACCEPTED JULY 1995